

АРТЕЕВ М. С.
ПРИМЕНЕНИЕ ПРОЕКТНЫХ ПАТТЕРНОВ ПРИ ПОСТРОЕНИИ
РАСШИРЯЕМОГО ROGUELITE-КОМПЛЕКСА

УДК 004.42:794.8, ГРНТИ 50.41.25

Статья поступила в редакцию 18.06.2026

Применение проектных паттернов при
построении расширяемого Roguelite-
комплекса

Applying design patterns when
building an extensible Roguelite
complex

М. С. Артеев

M. S. Arteev

Ухтинский государственный
технический университет, г. Ухта

Ukhta State Technical University,
Ukhta

В статье представлена работа по проектированию и разработке игрового программного комплекса Nightmare Hunters в жанре Roguelite для платформы Windows. Анализ предметной области показал, что Roguelite-игра строится на повторяемых попытках прохождения, бое в реальном времени и развитии персонажа в пределах попытки, а ведущие аналоги являются закрытыми коммерческими продуктами, на которых нельзя проверить собственные проектные решения. Для комплекса спроектирована слоистая архитектура с разделением игровой логики, данных и представления и реализован базовый игровой цикл: управление персонажем, боевые механики, награды и сохранение прогресса. Разработка собственного комплекса с расширяемой архитектурой позволяет наращивать состав персонажей, способностей, противников и комнат без переработки базовой логики и подтверждена функциональным и автоматизированным тестированием.

The article presents the work on the design and development of the Nightmare Hunters game software complex, a Roguelite for the Windows platform. The analysis of the subject area showed that a Roguelite game is built on repeatable play-through attempts, real-time combat and in-run character progression, while the leading analogues are closed commercial products on which one cannot verify one's own design decisions. A layered architecture separating game logic, data and presentation was designed for the complex, and the core gameplay loop was implemented: character control, combat mechanics, rewards and progress saving. Developing an in-house complex with an extensible architecture makes it possible to expand the set of characters, abilities, enemies and rooms without reworking the core logic, which was confirmed by functional and automated testing.

Ключевые слова: видеоигра, Roguelite, Nightmare Hunters, игровой программный комплекс, одиночный игровой процесс, игровая сессия, боевые механики, развитие персонажа, комнатное прохождение

Keywords: video game, Roguelite, Nightmare Hunters, game software suite, single-player gameplay, game session, combat mechanics, character progression, room-based progression

Введение

Разработка компьютерных игр относится к прикладным направлениям программной инженерии, где качество конечного продукта определяется не только визуальным исполнением, но и устойчивостью программной структуры. Для пользователя игра воспринимается через отзывчивое управление, корректную обработку боевых действий, понятную работу интерфейса, сохранение прогресса и предсказуемые переходы между игровыми состояниями.

Сегмент игр жанра Roguelite растёт и оценивается приблизительно в 4,3 миллиарда долларов в год [1]. При этом ведущие игры жанра являются коммерческими продуктами, их внутреннюю реализацию нельзя изучить или проверить, поэтому собственные проектные решения апробировать не на чем. Разработка собственного игрового программного комплекса с открытой расширяемой архитектурой вносит вклад в становление российского геймдева и примыкает к направлению интерактивных программных комплексов на игровых движках, например, VR-тренажёру правил безопасности в нефтяных шахтах, разработанному на кафедре [2].

Целью работы является разработка игрового программного комплекса Nightmare Hunters в жанре Roguelite для платформы Windows с настраиваемой сложностью прохождения комнат и гибкой системой модификации способностей и предметов. Для достижения поставленной цели решены задачи: проектирование масштабируемой архитектуры приложения, разработка боевой системы и искусственного интеллекта противников, реализация подсистемы способностей и статусных эффектов, системы экономики и инвентаря, генерации уровней и единого игрового цикла, а также тестирование и отладка программного продукта.

Предпроектное обследование

Игры жанра Action Roguelite объединяют бой в реальном времени и зависимость результата от действий игрока с сессионной структурой повторяемых попыток прохождения («забегов»), вариативностью, временным развитием персонажа в пределах попытки и метапрогрессом между попытками. Для таких игр важна не случайность сама по себе, а управляемое изменение игрового процесса: выбор комнаты, награды или усиления реализуется методами процедурной генерации контента и подчиняется алгоритмическим правилам [3]. В Nightmare Hunters процедурно формируется не геометрия уровня, а последовательность и связи игровых ситуаций, собираемых в граф комнат.

Предметная область находится на границе геймдизайна и программной инженерии: игра рассматривается как система правил и осмысленных действий пользователя [4].

Проблема разработки заключается в необходимости создать расширяемую структуру Roguelite-игры, в которой игровые правила, данные, интерфейс, боевые механики и сохранение прогресса не зависят напрямую от конкретных сцен и отдельных реализаций. Проект ограничивается одиночным локальным режимом; сетевой режим, серверная авторизация, онлайн-рейтинги и коммерческая монетизация в его состав не входят.

Игровой сценарий прохождения

Минимальный проверяемый сценарий Nightmare Hunters включает полный путь пользователя от запуска приложения до завершения попытки: ввод игрока вызывает действие, действие изменяет состояние игры, результат отображается через интерфейс, а значимые результаты переходят в сохраняемый прогресс. Основные этапы сценария приведены в Таблице 1.

Таблица 1. Основные этапы игрового сценария Nightmare Hunters

№	Этап	Содержание
1	Запуск приложения	Открывается главное меню, интерфейс принимает команды
2	Выбор или создание профиля	Игрок выбирает либо создаёт локальный профиль
3	Формирование забега	Фиксируются персонаж, сложность и улучшения, формируется исходное состояние
4	Управление персонажем	Перемещение, атака, способности и взаимодействия в реальном времени
5	Боевой эпизод	Действия игрока и противников изменяют здоровье, эффекты и состояние комнаты
6	Завершение комнаты	После выполнения условий формируется награда
7	Переход или завершение	Загружается следующая комната согласно маршруту, либо сессия завершается

Обзор аналогов

Для анализа выбраны три близких по жанру проекта: Wizard of Legend [8], Enter the Gungeon [9] и Soulstone Survivors [10]. Wizard of Legend важен системой активных способностей и комбинаций, Enter the Gungeon – комнатной структурой и циклом «бой – зачистка – награда», Soulstone Survivors – волновыми столкновениями и развитием персонажа внутри попытки. Сравнение приведено в Таблице 2.

Таблица 2. Сравнительный анализ аналогов

Критерий	Nightmare Hunters	Wizard of Legend	Enter the Gungeon	Soulstone Survivors
Жанровая основа	Roguelite	Подземелья, роуглайк	Подземелья, «пулевой ад»	Action Roguelite, арена
Боевая модель	Атаки, способности, столкновения	Магический бой, комбинации	Стрельба, уклонения, снаряды	Волны, навыки, синергия
Структура прохождения	Дискретные комнаты	Боевые зоны	Комнаты и этажи	Открытая арена, волны
Развитие персонажа	Усиления в пределах попытки	Заклинания, реликвии	Оружие, предметы	Навыки, оружие, метапрогресс

Ни один из проектов не является прямым образцом, а их закрытость не позволяет проверять собственные проектные решения. Поэтому от аналогов берутся отдельные жанровые ориентиры, а Nightmare Hunters разрабатывается как контролируемый Roguelite-цикл с возможностью дальнейшего расширения.

Результат разработки системы

При выборе структурной организации рассматривались монолитная сценарная структура на компонентах MonoBehaviour и слоистая архитектура. Первый вариант удобен на этапе раннего прототипирования, но при росте проекта привязка доменных правил к сценарным сценариям повышает связанность и подверженность каскадным ошибкам. Для Nightmare Hunters выбрана слоистая архитектура с разделением игровой логики, инфраструктурных сервисов, представления и данных, что соответствует принципу управления направлением зависимостей [6]. Выделены четыре области: Core (доменная логика – расчёт урона, здоровье, эффекты, способности, комнаты, награды, инвентарь), Infrastructure (ввод, сохранения, настройки, локализация, загрузка сцен), Presentation (MonoBehaviour-адаптеры, HUD, визуальные эффекты, коллайдеры, префабы) и Installers (композиция зависимостей). Слой Core не обращается к иерархии сцены и не создаёт префабы; объекты представления получают данные от сервисов и передают команды обратно, но не становятся источником правил. Логическая архитектура приведена на Рисунке 1.

Связывание компонентов выполняется через внедрение зависимостей (Zenject) на трёх точках композиции: ProjectInstaller (сервисы, сохраняемые между сценами), MenuInstaller (главное меню) и GameInstaller (зависимости игровой сцены). В подсистемах применён ряд проектных решений, обобщённых в Таблице 3 [5, 7].

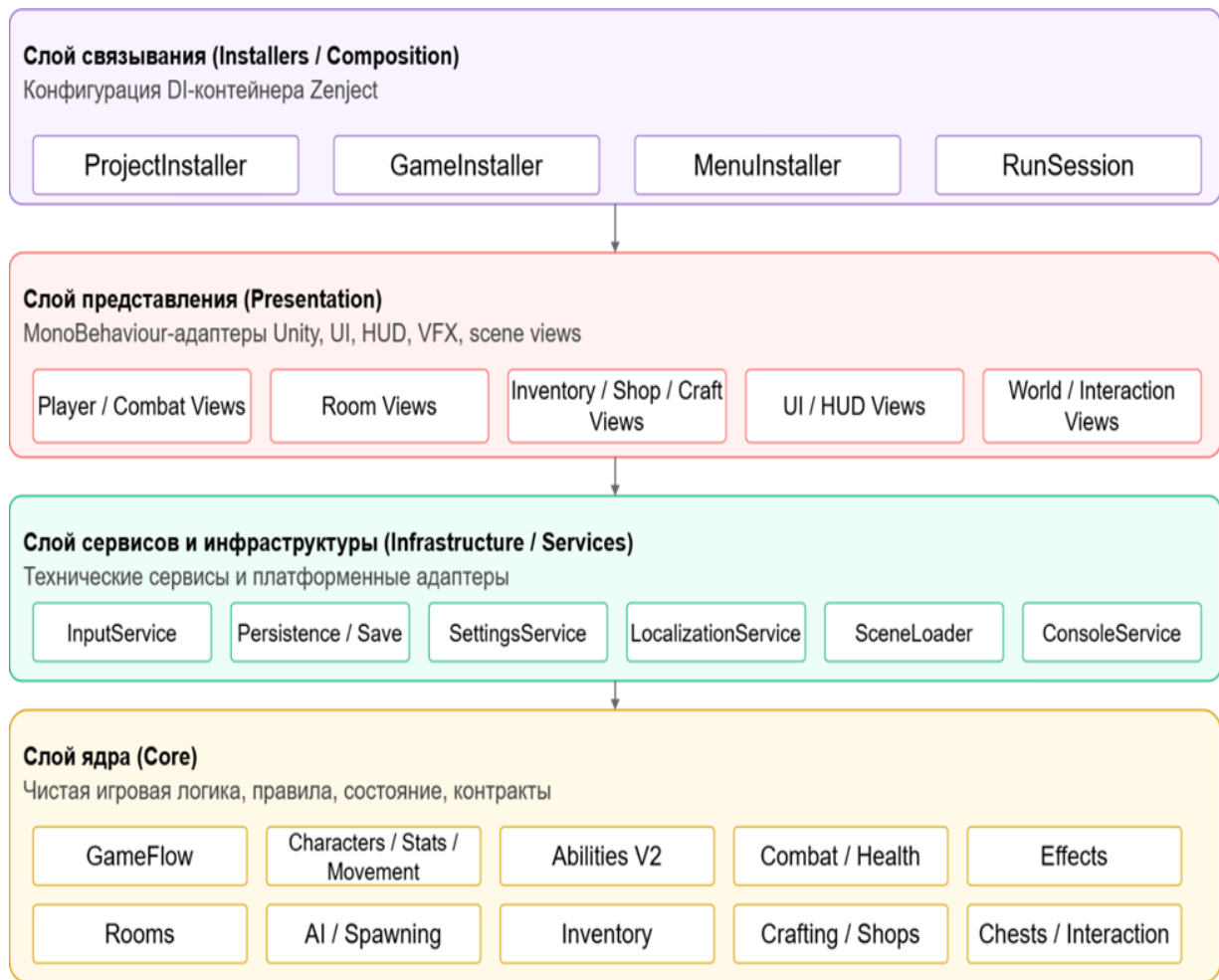


Рисунок 1. Логическая архитектура

Таблица 3. Проектные решения подсистем Nightmare Hunters

Область	Проектное решение	Назначение
1	2	3
Действия персонажа	Command	Единое представление атак, способностей и взаимодействий
Способности	Request	Передача сценовых запросов из Core в Presentation без зависимости от префабов
Противники	Strategy	Выбор поведения через профили и стратегии искусственного интеллекта
Жизненный цикл приложения	State	Управляемые переходы между состояниями (меню, загрузка, забег, поражение, победа)
Здоровье, смерть, награды	Observer	Реакция подсистем на изменение состояния без жёстких прямых связей
Игровой контент	Data-Driven Design	Задание способностей, эффектов, предметов, комнат и противников через данные
Доступ к данным	Repository / Storage Adapter	Изоляция способа хранения от игровой логики

Вариативность забега обеспечивается процедурной сборкой маршрута из заранее подготовленных комнат. Генератор RoomRunMapGenerator формирует абстрактный граф забега на слое Core, без зависимости от сценых объектов Unity. В текущей конфигурации маршрут содержит 12 небоссовых комнат и завершается боссом, узлы распределяются по 4 дорожкам, на выходе предлагаются до 2 переходов, 3 комнаты гарантированы, на каждый путь приходится не менее 6 боевых и не более 2 элитных комнат. Параметры вынесены в сериализуемые определения, что позволяет менять длину, число дорожек и состав комнат без правки кода [3]. Генерация детерминирована: явное зерно (seed) и состояние генератора случайных чисел включаются в данные сохранения RunSaveData, что позволяет восстановить незавершённый забег с тем же маршрутом. Диаграмма генератора графа приведена на Рисунке 2.



Рисунок 2. Генератор графа маршрута забега

Информационная база не строится на выделенной СУБД. Статические игровые данные (персонажи, способности, эффекты, противники, комнаты, предметы, награды, рецепты, магазины) задаются ScriptableObject-ресурсами и связываются через централизованные реестры, что соответствует подходу, управляемому данными. Каждая способность задаётся ресурсом

GameplayAbilityDefinition и обрабатывается общим экземпляром GameplayAbilityInstance, который проверяет перезарядку, заряды и состояние персонажа, но не создаёт объекты сцены напрямую: сценовое исполнение делегируется обработчикам уровня Presentation через типизированные запросы. Добавление новой способности или предмета не требует изменения логики сцены и повторной компиляции проекта.

Для боя в реальном времени критична корректная маршрутизация ввода. Реализована стековая блокировка игровых действий InputLockScope: открытие модального окна (инвентарь, экран награды, пауза) помещает блокировку в стек, и пока стек непуст, сервис ввода InputService отдаёт пустой снимок, а команды боя и перемещения не доходят до игровой логики. Это исключает ситуацию, когда одно нажатие одновременно выбирает элемент интерфейса и активирует действие персонажа.

Локальное сохранение защищено от сбоя и подмены. Запись выполняется атомарно: данные помещаются во временный файл, прежняя версия переводится в резервную копию, и только затем временный файл переименовывается в основной. Содержимое сохранения защищено криптоконвертом: применяется симметричное шифрование AES-256 в режиме CBC с дополнением PKCS7, а подлинность проверяется HMAC-SHA256. Ключи шифрования и аутентификации разделены и вырабатываются по схеме, аналогичной HKDF, с учётом фиксированной соли, версии структуры и вида сохранения; при каждой записи создаётся новый случайный 16-байтовый вектор инициализации. В расчёт тега добавляются связанные данные (сигнатура BNH_SAVE, тип и версия сохранения), что предотвращает атаки замещения, а сравнение тегов выполняется за фиксированное время. Любое изменение метаданных, зашифрованного блока или тега делает загрузку невозможной; при обнаружении вмешательства система переходит в безопасное состояние без аварийного завершения. Внешний вид игрового процесса и экрана результата приведён на Рисунках 3 и 4.

Работоспособность демонстрационной версии проверена двумя способами. Модульное тестирование (Unity Test Framework, режим EditMode) охватило шесть групп компонентов: боевую логику (DamageSystem, HealthSystem, DamageContext), эффекты и модификаторы (EffectSystem), игровой ввод (InputService, InputLockScope), комнаты и награды (RoomRoutePlanner и сервисы наград), инвентарь и предметы (InventorySystem, ItemDropService) и сохранение данных (ProfileService, RunSession, JsonRunSaveStorage); все группы выполнили ожидаемые операции без критических ошибок. Функциональное и интеграционное тестирование выполнялось по пользовательским сценариям; результаты приведены в Таблице 4.

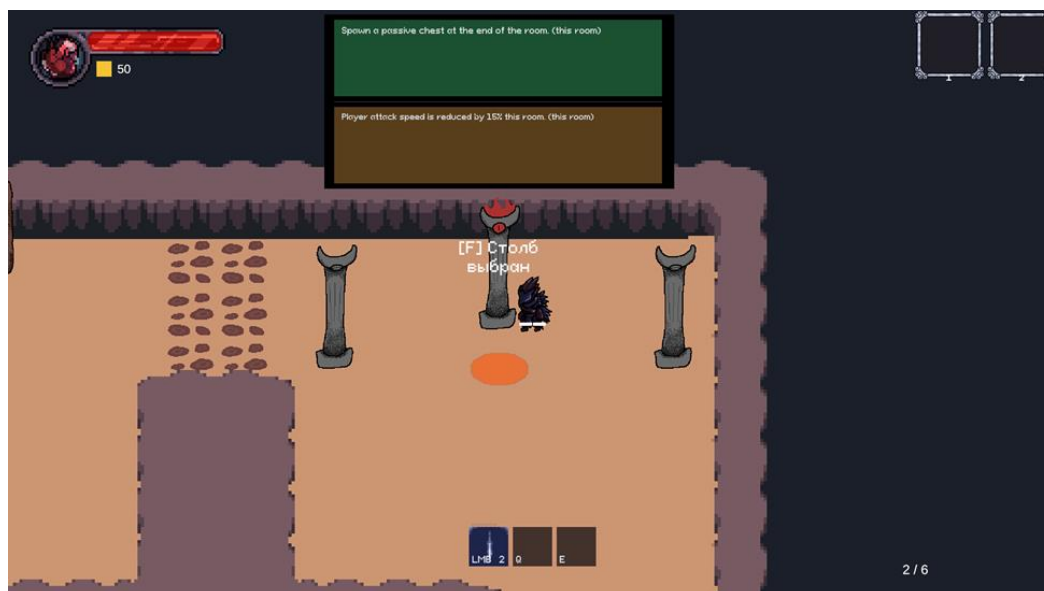


Рисунок 3. Игровой процесс Nightmare Hunters

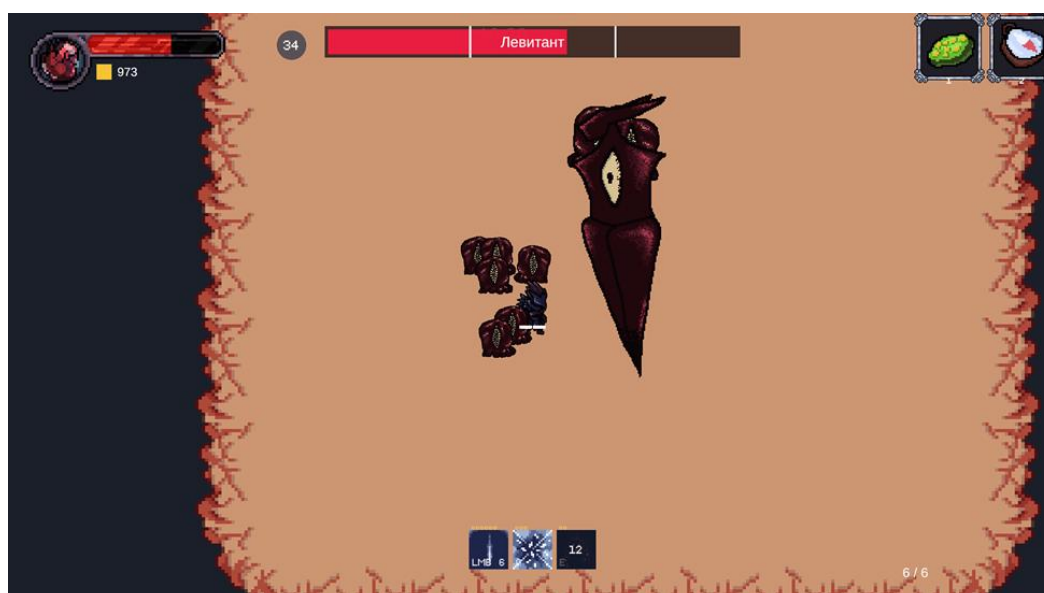


Рисунок 4. Бой с боссом в конце прохождения

Таблица 4. Результаты функционального тестирования

№	Проверяемый сценарий	Фактический результат	Статус
1	2	3	4
1	Запуск приложения	Главное меню загружается штатно	Пройден
2	Создание локального профиля	Профиль создаётся и сохраняется локально	Пройден
3	Выбор существующего профиля	Профиль применяется при запуске прохождения	Пройден
4	Запуск новой игровой сессии	Сцена RoomModeMVP загружается, сессия инициализируется	Пройден

Продолжение Таблицы 4

1	2	3	4
5	Управление персонажем	Перемещение, атака и действия выполняются корректно	Пройден
6	Боевое взаимодействие	Противники получают урон, здоровье обновляется	Пройден
7	Завершение боевого эпизода	Завершение комнаты фиксируется системой	Пройден
8	Получение награды	Награда отображается и применяется к состоянию	Пройден
9	Переход к следующей комнате	Переход выполняется через интерактивный объект	Пройден
10	Работа интерфейсных режимов	Команды боя и перемещения блокируются при модальном окне	Пройден
11	Сохранение локального прогресса	Данные доступны при повторном запуске	Пройден
12	Завершение игровой сессии	Сессия завершается без зависания и неопределённого состояния	Пройден

Все двенадцать функциональных сценариев пройдены. Отдельно подтверждены два критичных для жанра сценария: при открытых интерфейсных окнах игровой ввод корректно блокируется, а проверка сохранения через создание профиля, изменение состояния прохождения и повторное обращение к данным показала восстановление значимых данных без ручного редактирования файлов и без аварийного завершения приложения.

Заключение

Разработан игровой программный комплекс Nightmare Hunters в жанре Roguelite для платформы Windows, реализующий базовый цикл одиночного прохождения: управление персонажем, боевые взаимодействия, получение наград, переходы между комнатами и сохранение прогресса. Слоистая архитектура и подход, управляемый данными, позволяют расширять состав персонажей, способностей, противников и комнат без переработки базовой логики; детерминированная процедурная генерация обеспечивает вариативность и восстановление забега, а защита сохранений на основе AES-256-CBC и HMAC-SHA256 обеспечивает обнаружение подмены и устойчивость к сбоям. Работоспособность подтверждена функциональным и автоматизированным тестированием.

Дальнейшее развитие проекта связано с расширением набора персонажей, противников, способностей, предметов и комнат, балансировкой системы наград, улучшением визуального и звукового сопровождения и расширением набора автоматизированных тестов.

Список использованных источников и литературы

1. DataIntelto. Roguelite Games Market Research Report // DataIntelto. 2026. URL: <https://dataintelto.com/report/roguelite-games-market> (дата обращения: 20.05.2026).
2. Болотов И. Г., Рочев К. В. VR-тренажер правил безопасности в нефтяных шахтах // Информационные технологии в управлении и экономике. 2022. № 1. С. 77–86. URL: <http://itue.ru/Issue/Article/194>.
3. Shaker N., Togelius J., Nelson M. J. Procedural Content Generation in Games. Cham: Springer, 2016. 237 p. DOI: 10.1007/978-3-319-42716-4.
4. Salen K., Zimmerman E. Rules of Play: Game Design Fundamentals. Cambridge: MIT Press, 2003. 688 p.
5. Nystrom R. Game Programming Patterns. 2014. URL: <https://gameprogrammingpatterns.com/> (дата обращения: 26.04.2026).
6. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Boston: Prentice Hall, 2018. 432 p.
7. Fowler M. Patterns of Enterprise Application Architecture. Boston: Addison-Wesley Professional, 2002. 560 p.
8. Wizard of Legend // Contingent99: press kit. URL: https://wizardoflegend.com/press/wizard_of_legend/index.html (дата обращения: 30.05.2026).
9. Enter the Gungeon // Devolver Digital. URL: <https://www.devolverdigital.com/games/enter-the-gungeon> (дата обращения: 30.05.2026).
10. Soulstone Survivors // Steam. URL: https://store.steampowered.com/app/2066020/Soulstone_Survivors/ (дата обращения: 30.05.2026).

List of references

1. DataIntelto. Roguelite Games Market Research Report. DataIntelto. 2026. Available at: <https://dataintelto.com/report/roguelite-games-market> (accessed 20.05.2026).
2. Bolotov I.G., Rochev K.V. VR simulator of safety rules in oil mines. Information Technologies in Management and Economics. 2022. No. 1. Pp. 77–86. Available at: <http://itue.ru/Issue/Article/194>.
3. Shaker N., Togelius J., Nelson M.J. Procedural Content Generation in Games. Cham: Springer, 2016. 237 p. DOI: 10.1007/978-3-319-42716-4.
4. Salen K., Zimmerman E. Rules of Play: Game Design Fundamentals. Cambridge: MIT Press, 2003. 688 p.
5. Nystrom R. Game Programming Patterns. 2014. Available at: <https://gameprogrammingpatterns.com/> (accessed 26.04.2026).
6. Martin R.C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Boston: Prentice Hall, 2018. 432 p.
7. Fowler M. Patterns of Enterprise Application Architecture. Boston: Addison-Wesley Professional, 2002. 560 p.

8. Wizard of Legend. Contingent99: press kit. Available at: https://wizardoflegend.com/press/wizard_of_legend/index.html (accessed 30.05.2026).

9. Enter the Gungeon. Devolver Digital. Available at: <https://www.devolverdigital.com/games/enter-the-gungeon> (accessed 30.05.2026).

10. Soulstone Survivors. Steam. Available at: https://store.steampowered.com/app/2066020/Soulstone_Survivors/ (accessed 30.05.2026).